REMARKS/ARGUMENTS

The amendments made herein are in response to the Office Action of September 22, 2003 (Final Office Action).

Claims 1-2, 4-5, 9-10, and 12-13 have been rejected under 35 U.S.C. § 103(a), as being unpatentable over a Riehle, Dirk [1996] "The Event Notification Pattern: Integrating Implicit Invocation with Object-Orientation" in Theory and Practice of Object Systems, 2, 1, pages 43-52 (Riehle).

Prior to turning to the rejections of the art, a synopsis of the Applicants' invention is in order. The Applicants' invention is a generalized solution for adding remote event handling capabilities within a distributed environment. The remote event handling capabilities of the Applicants' invention are achieved by (A) establishing a Notifier object in a client application that can detect client events, (B) establishing a Listener object in a server application that is linked to the Notifier, and (C) linking a method located remotely from the server to the Listener object of the server. Accordingly, an event detected in a client (A) notifies a process in a server (B) that fires a method from a location remote from the server (C). This arrangement permits a client program to generically trigger remotely located methods via a server intermediary even though the client program is unaware of the actual location of the remotely called method. Claim 1 of the Applicants' invention has been included below to verify the above characterization of the Applicants' invention. (Letters A, B, and C, bold, and italics added)

> 1. A method for establishing a location transparent event handler comprising the steps of:
>
> ***(A)*** establishing a **Notifier object in a client application** for execution in a first process address space, said Notifier object based upon a Notifier class, said Notifier object having a list of Listener objects to be notified upon an event occurrence;
>
> ***(B)*** establishing a **Listener object in a server application** for execution in a second process address space separate from said first process address space, said Listener object based upon a Listener class, said Listener object

defining a method to be called upon the occurrence of said event, **said Listener object enabled to be callable from said Notifier object**; and,

(C) generating a Listener object stub for said Listener object, said Listener object stub configured to be added to said list of Listener objects in said Notifier object, said Listener object stub further **configured to remotely call said defined method in said Listener object in response to receiving notification of an event** from said Notifier object,

whereby upon said event occurrence, said Notifier object can traverse said list of Listener objects and can notify said Listener object stub of said event occurrence thereby creating a remote call to said defined method in said Listener object.

Riehle teaches an event notification pattern for implicitly invoking events to maintain state relationships. In the Office Action on page 4, the Examiner acknowledges that "Riehle does not teach an instance of a Notifier class in a first address space and an instance of an instance of a Listener object in a second address space." It has been alleged, however, that such an arrangement is suggested by the IACEventLink object. Specifically, the forwarding of event notification from a Notifier to a Listener (from A to B) can be performed using the IACEventLink as noted in page 8 of Riehle.

It should be noted that the purpose of Riehle, maintaining state relationships in an application, can be fully satisfied by proceeding from step A to B. Riehle provides no motivation, teachings, or suggestions for performing steps B to C after steps A to B. Further, Riehle does not teach or suggest performing steps B to C at all. Therefore, the Applicants' invention is not obvious in view of Riehle.

Applicants herein emphasize the relevance of placing the Notifier in a client and the Listener in a server, which is not taught by Riehle, is significant when proceeding from A to B to C as claimed in the Applicants' invention. If the Applicants' invention only disclosed steps A to B, then the assertion that it is obvious to place the Notifier in a client and the Listener in a server may have merit since such a placement would be somewhat arbitrary and not be especially significant to the invention as claimed. The advantages of the placement of the Notifier in the client and the Listener in the server, as asserted in the previous response, become apparent when remotely triggering a

procedure from a server using a method such as Remote Method Invocation (RMI). Thus, the placement becomes significant when proceeding from A to B to C.

Regarding claim 4, the claim discloses a method where an event detected in a first processing address space (A) notifies a process in a second process address space (B) that fires a method from a location remote from the second process address space (C). While Riehle debatably teaches step A to B, Riehle provides no motivation, teachings, or suggestions for performing steps B to C after steps A to B. Further, Riehle does not teach or suggest performing step B to C. Claim 4 of the Applicants' invention has been included below to verify the above characterization of the Applicants' invention. (Letters A, B, and C, bold, and italics added)

> 4. A method for performing location transparent event handling comprising the steps of:
>
> ***(A)*** creating an instance of a **Notifier class in a first process address** space, said Notifier instance having a list of Listener objects to be notified upon an event occurrence;
>
> **(B)** creating an instance of a **Listener class in a second process address** space, said Listener instance having a method to be called upon the occurrence of said event, said Listener instance enabled to be callable from said Notifier instance, wherein said Notifier instance and said Listener instance are configured to perform location transparent event handling;
>
> ***(C)*** inserting a Listener object stub in said list of Listener objects in said Notifier instance in said first process address space, said Listener object stub **configured to remotely call said defined method** in said Listener instance;
>
> receiving an event occurrence in said Notifier instance; and,
>
> responsive to receiving said event occurrence, traversing said list of Listener objects, passing said event to said Listener object stub, creating in said Listener object stub a remote call to said defined method in said Listener instance, and executing said defined method in said Listener instance.

Regarding claims 2, 5, 10, and 13, an assertion is made that "one of ordinary skill in the art knows the advantages of Java programming language wherein the language is neutral platform and Java Virtual Machine is needed in each system in order to execute the Java programs and Java is used in distributed applications." Applicants are uncertain as to what supporting material has used to base this allegation upon as no

reference has been provided. As Applicants disagree with the allegation as applied to the specific invention disclosed herein as stated previously in the first Office Action response, supporting material is again requested.

Applicants respectfully submit that the prior art of record fails to provide any teachings or suggestion that Applicants' invention as claimed absent Applicants' own disclosure. Further, no known art references express the desirability to combine Applicants' teachings with teachings of the Java programming language and/or Java Virtual machines. As noted in *In re Gordon*, 733 F.2d 099, 221 USPQ at 1127 "the mere fact that prior art may be modified in the manner suggested by the Examiner does not make the modification obvious unless the prior art suggests the desirability of the modification."

Claims 3, 6-8, 11, and 14-16 have been rejected under 35 U.S.C. § 103(a), as being unpatentable over Riehle in view of OMG TC Document 95.8.19 [1995] "COM/CORBA Interworking RFP Part A" (OMG) and in further view of "Sun Microsystems, Inc. Remote Method Invocation Specification" (Sun RMI).

Regarding the Sun RMI reference, Applicants have specifically claimed that one method of performing step B to C is to use RMI and the RMI registry as officially specified by Sun RMI. Sun RMI provides no teachings or suggestions regarding step A to B as disclosed by Applicants' invention. Further, Sun RMI does not provide any teachings or suggestions concerning the desirability to combine steps B and C with steps A and B as taught by Applicants' invention.

OMG officially defines CORBA objects as objects that can perform many of the same functions as preformed by the Sun RMI framework. No motivation to combine the OMG reference with the teachings of Riehle has been shown other than an assertion that such a combination is possible. As previously noted, "the mere fact that prior art may be modified in the manner suggested by the Examiner does not make the modification obvious unless the prior art suggests the desirability of the modification." No such desirability has been suggested by either Riehle, OMG, Sun RMI, or any other cited reference, or combination thereof.

Further, Applicants disagree with the statement on page 8 of the Office Action that "one of ordinary skill in the art knows that RMI generates stub that executes on the client side, and RMI stub equivalent to Corba proxy." CORBA and RMI are very different protocols that function in different ways as detailed within the first Office Action response. Functions and features available under one model are different from functions and features available under the other model. No teachings in the art show how to reconcile these two different models to permit functions conventionally absent from the RMI model to be supplemented with functions conventionally available via the CORBA model. The two models and functions contained therein are not interchangeable, thereby causing one of ordinary skill in the art to select one model over the other depending upon the task being performed by a software developer.

In light of the above discussion, neither Riehle, OMG, Sun RMI, nor any combination thereof teach or suggest Applicants' invention as claimed, namely a method from A to B to C. Therefore, withdrawal of the 35 U.S.C. § 103(a) rejection of claims 1-16 is respectfully requested. Applicants believe that this application is now in full condition for allowance, which action is respectfully requested. Applicants request that the Examiner call the undersigned if clarification is needed on any matter within this Amendment, or if the Examiner believes a telephone interview would expedite the prosecution of the subject application to completion.

Respectfully submitted,

Date: 22 Dec 2003

Gregory A. Nelson, Registration No. 30,577
Kevin T. Cuenot, Registration No. 46,283
Brian K. Buchheit, Registration No. 52,667
AKERMAN SENTERFITT
Post Office Box 3188
West Palm Beach, FL 33402-3188
Telephone: (561) 653-5000